Pokročilé operace na stromech

Příklad 1: Pokud si ve vnitřních vrcholech stromů pamatujeme různé hodnoty (počet prvků, maximum, ...), navrhněte, jak je udržovat během vyvažování.

Příklad 2: Mějme vyhledávací strom ukládající čísla řazená dle klíčů. Navrhněte operaci intervalového shiftu, která všechny hodnoty na intervalu klíčů posune hodnoty o δ v čase $\mathcal{O}(\log n)$.

Hint: Není možné skutečně změnit všechny hodnoty, použijeme metodu líného provědění operací

(+) Kterou operaci z minulého cvičení nelze implementovat na stromě, který umí intervalový shift?

(a,b)-stromy

Příklad 3: Zkuste rozmyslet, proč nastavení $b = 2a - 1$ bude mít horší vlastnosti než $b = 2a$. (Speciálně $(2, 4)$ jsou efektivní, ale $(2, 3)$ nejsou)

Příklad 4: Určete minimální a maximální počet prvků v (a,b)-stomě hloubky h. Ukažte, že kořen musí mít povoleno mít méně než a synů.

Příklad 5: Navrhněte Merge a Split pro (a,b)-stromy v logaritmickém čase. Merge slévá dva stromy (jeden s menšími a druhý s většími hodnotami), Split dostane pivotní hodnotu a rozštípne strom na dva (jeden s menšími a druhý s většími hodnotami).

Příklad 6: (+)Trojitá fůze Upravme (a,b)-stromy tak, aby při fůzi spojovaly tři sousední uzly (namísto dvou), a při štěpení rozštěpily dva sousední uzly na tři (místo jednoho na dva). Najděte přesné podmínky pro hodnoty a, b pro které takové schéma funguje.

(+) O kolik se zlepšilo využití paměti (pokud uzly implementujeme jako pole velikosti b)?

Amortizace

Příklad 7: Líné vyvažování (kapitola 9.4. v Průvodci) Definujme BVS, kde si v každém vrcholu udržujeme počet prvků v podstromě. Vrchol je vyvážený, pokud oba podstromy obsahují maximálně 2/3 celkového počtu prvků. Pokud je tato podmínka (při Insert či Delete) porušena, najdeme nejvyšší uzel ve kterém neplatí, a celý jeho podstrom perfektně vyvážíme přebudováním.

Definujme potenciál vrcholu jako (absolutní) rozdíl ve velikostech podstromů, potenciál stromu je součet potenciálů vrcholů (nebo 0 pokud je rozdíl ≤ 1)

- Ukažte, že hloubka stromu je $\mathcal{O}(\log n)$
- Ukažte, že Insert a Delete zvýší potenciál stromu nejvýše o $\mathcal{O}(\log n)$
- Formálně ukažte, že amortizovaná složitost operací Insert a Delete je $\mathcal{O}_a(\log n)$ (amortizovaná složitost je definovaná jako skutečná složitost plus změna potenciálu)

Domácí úkol 6:

Navrhněte úpravu nějakého vyhledávacího stromu tak aby kromě Find, Insert a Delete (v $\mathcal{O}(\log n)$) uměl i následující operace:

- Median() - vrátí medián všech hodnot v čase $\mathcal{O}(1)$. Medián na lichém počtu prvků je definován jako prostřední prvek, na sudém počtu je to průměr prostředních dvou.
- IntAvg(a,b) - vrátí průměr všech hodnot z intervalu $< a, b >$ v čase $\mathcal{O}(\log n)$.

Hint: Pro dotazy fungující v čase $\mathcal{O}(1)$ je potřeba aktuální hodnotu "nějak" průběžně udržovat. Pro intervalové dotazy v čase $\mathcal{O}(\log n)$ je potřeba udržovat aktuální hodnoty v podstromech podobně jako na cvičení (inspiraci můžete najít v kapitole 4.5. v Průvodci).

Předpokládáme, že strom ukládá celá čísla, řazená dle hodnot (bez klíčů). Chcete-li, můžete si představovat konkrétní typ vyhledávacího stromu (AVL, (a,b), RB, ...). Pro jednoduchost mějme hodnoty pouze v listech. Navíc se nám může hodit trik ze cvičení, kde si listy propojíme do spojového seznamu dle pořadí (a seznam udržujeme při Insert a Delete).

Operace chceme implementovat tak, aby byly přesné. Počítače neumí operovat s desetinnými čísly (můžete otestovat, že téměř vždy $0.8 + 0.2 \neq 1$).

Chceme zachovat složitost základních operací.

Popisovat můžete slovně, není vyžadován pseudokód. V řešení popište změny v reprezentaci stromu, implementaci příslušných operací, a změny v základních operacích (Find,Insert,Delete), které je třeba provést. Nezapomeňte také na úpravy ve vyvažování! Stručně zdůvodněte časovou složitost.